```
FFFFFFFFFFFFFFF     000000000     RRRRRRRRRRR      RRRRRRRRRRR      TTTTTTTTTTTTTTT   LLL
FFFFFFFFFFFFFFF     000000000     RRRRRRRRRRR      RRRRRRRRRRR      TTTTTTTTTTTTTTT   LLL
FFFFFFFFFFFFFFF     000000000     RRRRRRRRRRR      RRRRRRRRRRR      TTTTTTTTTTTTTTT   LLL
FFF               000      000    RRR       RRR    RRR       RRR         TTT          LLL
FFF               000      000    RRR       RRR    RRR       RRR         TTT          LLL
FFF               000      000    RRR       RRR    RRR       RRR         TTT          LLL
FFF               000      000    RRR       RRR    RRR       RRR         TTT          LLL
FFF               000      000    RRR       RRR    RRR       RRR         TTT          LLL
FFFFFFFFFFF       000      000    RRRRRRRRRRR      RRRRRRRRRRR           TTT          LLL
FFFFFFFFFFF       000      000    RRRRRRRRRRR      RRRRRRRRRRR           TTT          LLL
FFFFFFFFFFF       000      000    RRRRRRRRRR       RRRRRRRRRR            TTT          LLL
FFF               000      000    RRR  RRR         RRR  RRR             TTT          LLL
FFF               000      000    RRR  RRR         RRR  RRR             TTT          LLL
FFF               000      000    RRR   RRR        RRR   RRR            TTT          LLL
FFF               000      000    RRR    RRR       RRR    RRR           TTT          LLL
FFF               000      000    RRR     RRR      RRR     RRR          TTT          LLL
FFF               000      000    RRR      RRR     RRR      RRR         TTT          LLL
FFF                 000000000     RRR       RRR    RRR       RRR        TTT          LLLLLLLLLLLLLLL
FFF                 000000000     RRR        RRR   RRR        RRR       TTT          LLLLLLLLLLLLLLL
FFF                 000000000     RRR        RRR   RRR        RRR       TTT          LLLLLLLLLLLLLLL
```

```
FFFFFFFFF   000000   RRRRRRR   FFFFFFFFF  MM     MM  TTTTTTTTTT  IIIIII   NN     NN  TTTTTTTTTT
FFFFFFFFF   000000   RRRRRRR   FFFFFFFFF  MM     MM  TTTTTTTTTT  IIIIII   NN     NN  TTTTTTTTTT
FF          00    00 RR    RR  FF         MMMM MMMM      TT         II    NN     NN      TT
FF          00    00 RR    RR  FF         MM MMM MM      TT         II    NNNN   NN      TT
FF          00    00 RR    RR  FF         MM  MM  MM      TT         II    NNNN   NN      TT
FFFFFFFF    00    00 RRRRRRR   FFFFFFF    MM     MM      TT         II    NN NN  NN      TT
FFFFFFFF    00    00 RRRRRRR   FFFFFFF    MM     MM      TT         II    NN  NN NN      TT
FF          00    00 RR  RR    FF         MM     MM      TT         II    NN   NNNN      TT
FF          00    00 RR   RR   FF         MM     MM      TT         II    NN   NNNN      TT
FF          00    00 RR    RR  FF         MM     MM      TT         II    NN     NN      TT
FF          000000   RR     RR FF         MM     MM      TT      IIIIII   NN     NN      TT
FF          000000   RR     RR FF         MM     MM      TT      IIIIII   NN     NN      TT

LL          IIIIII      SSSSSSSS
LL          IIIIII      SSSSSSSS
LL            II      SS
LL            II      SS
LL            II      SS
LL            II      SS
LL            II         SSSSSS
LL            II         SSSSSS
LL            II              SS
LL            II              SS
LL            II              SS
LL            II              SS
LLLLLLLLLL  IIIIII      SSSSSSSS
LLLLLLLLLL  IIIIII      SSSSSSSS
```

```
    1    0001  0  MODULE FOR$$FMT_INTRP (%TITLE'Fortran Format Statement Interpreter'
    2    0002  0                        -IDENT = '2-037'          ! File: FORFMTINT.B32  Edit: SBL2037
    3    0003  0                        ) =
    4    0004  1  BEGIN
    5    0005  1
    6    0006  1  !*******************************************************************************
    7    0007  1  !*                                                                             *
    8    0008  1  !*   COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                   *
    9    0009  1  !*   DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                    *
   10    0010  1  !*   ALL RIGHTS RESERVED.                                                      *
   11    0011  1  !*                                                                             *
   12    0012  1  !*   THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED     *
   13    0013  1  !*   ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE     *
   14    0014  1  !*   INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER     *
   15    0015  1  !*   COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY     *
   16    0016  1  !*   OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY     *
   17    0017  1  !*   TRANSFERRED.                                                              *
   18    0018  1  !*                                                                             *
   19    0019  1  !*   THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE     *
   20    0020  1  !*   AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT     *
   21    0021  1  !*   CORPORATION.                                                              *
   22    0022  1  !*                                                                             *
   23    0023  1  !*   DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS     *
   24    0024  1  !*   SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                   *
   25    0025  1  !*                                                                             *
   26    0026  1  !*                                                                             *
   27    0027  1  !*******************************************************************************
   28    0028  1
   29    0029  1  !++
   30    0030  1  !  FACILITY: FORTRAN
   31    0031  1  !
   32    0032  1  !  ABSTRACT:
   33    0033  1  !
   34    0034  1  !      This module interprets FORTRAN format statements
   35    0035  1  !      which have been pre-compiled into an encoded form by either the
   36    0036  1  !      FORTRAN compiler or the run-time format compiler,
   37    0037  1  !      FOR$$FMT_COMPIL.  It is independent of READ and WRITE semantics
   38    0038  1  !      and is executed at both the READ Formatted and WRITE Formatted
   39    0039  1  !      User Data Formatters (UDF) level of abstraction.
   40    0040  1  !
   41    0041  1  !  AUTHOR: Peter Yuo, CREATION DATE: 17-Feb-77
   42    0042  1  !
   43    0043  1  !  MODIFIED BY:
   44    0044  1  !      Peter Yuo, 25-Feb-77, Version 1
   45    0045  1  !  01    -       Original
   46    0046  1  !
   47    0047  1  !      Richard Grove, 19-Aug-77, Version 2
   48    0048  1  ! [Previous edit history removed.  SBL 23-Aug-1982]
   49    0049  1  ! 2-032 - Add defaults for O and Z format width when value is not 1, 2, 4, 8 or
   50    0050  1  !          16 bytes.  SBL 29-Dec-1980
   51    0051  1  ! 2-033- Improved fix for 2-032, courtesy of Joel CLinkenbeard.  SBL 8-Jan-1981
   52    0052  1  ! 2-034 - Convert FOR$$FMT_INTRP1 to JSB linkage for better performance.
   53    0053  1  !          JAW 29-Jul-1981
   54    0054  1  ! 2-035 - Miscellaneous performance enhancements:  JAW 29-Jul-1981
   55    0055  1  !          Check for certain specific one-byte format codes at the outset
   56    0056  1  !            and special-case them.
   57    0057  1  !          For all format codes, if optional second byte is not present,
```

```
:    58          0058  1 |           bypass checks for VFEs and for optional forms of RC and W.
:    59          0059  1 |         Break FI_ACT into two tables, each having 1-byte entries,
:    60          0060  1 |           placing the special action in FI_ACT_2 and indicating the
:    61          0061  1 |           need for special action with the low-order bit of FI_ACT.
:    62          0062  1 |           Select a special action only if this bit is set.
:    63          0063  1 |         For codes _DF through _DD, check for element size of 4 first.
:    64          0064  1 |         Narrow the scope of ACT, FMT_REPR and P, which are not needed
:    65          0065  1 |           in the outermost block, to conserve registers.
:    66          0066  1 |         Replace CASE on V_RC_TYPE with IF ... THEN to avoid an EXTZV.
:    67          0067  1 | 2-036 - Correct range check of P value from VFEs.  SBL 23-Aug-1982
:    68          0068  1 | 2-037 - Allow zero-value VFEs for W, D and E fields only.  Use prologue
:    69          0069  1 |         file.  SBL 26-Apr-1983
:    70          0070  1 |-
```

```
  72     0071  1  !
  73     0072  1  !  PROLOGUE FILE:
  74     0073  1  !
  75     0074  1
  76     0075  1  REQUIRE 'RTLIN:FORPROLOG';              ! FORTRAN Definitions
  77     0141  1  SWITCHES ZIP;                           ! Optimize for speed
  78     0142  1
  79     0143  1  !
  80     0144  1  !  TABLE OF CONTENTS:
  81     0145  1  !
  82     0146  1
  83     0147  1  FORWARD ROUTINE
  84     0148  1      FOR$$FMT_INTRP0 : JSB_FMT0 NOVALUE,    ! initialization
  85     0149  1      FOR$$FMT_INTRP1 : JSB_FMT1 NOVALUE;    ! Interpret until a data format code
  86     0150  1
  87     0151  1  !
  88     0152  1  !  MACROS:
  89     0153  1  !
  90     0154  1        NONE
  91     0155  1  !
  92     0156  1  !  EQUATED SYMBOLS:
  93     0157  1  !
  94     0158  1        NONE
  95     0159  1  !
  96     0160  1  !  OWN STORAGE:
  97     0161  1  !
  98     0162  1        NONE
  99     0163  1  !
 100     0164  1  !
 101     0165  1  !  EXTERNAL REFERENCES:
 102     0166  1  !
 103     0167  1
 104     0168  1  EXTERNAL ROUTINE
 105     0169  1      FOR$$SIGNAL_STO : NOVALUE,           ! Signal_stop FOR$_abcmnoxyz, given
 106     0170  1                                          ! (short) Fortran error number (FOR$K_abcmnoxyz)
 107     0171  1                                          ! as a parameter
 108     0172  1      FOR$$SIGNAL : NOVALUE;              ! Signal FOR$_abcmnoxyz, given (short)
 109     0173  1
 110     0174  1                                          ! FORTRAN error number (FOR$K_abcmnoxyz)
 111     0175  1                                          ! as a parameter.
 112     0176  1
```

FOR$$FMT_INTRP  Fortran Format Statement Interpreter
2-037

F 13
16-Sep-1984 00:25:18    VAX-11 Bliss-32 V4.0-742
14-Sep-1984 12:32:00    [FORRTL.SRC]FORFMTINT.B32;1

Page  4
(3)

```
:  114    0177  1  GLOBAL ROUTINE FOR$$FMT_INTRPO                    ! Format interpreter initialization
:  115    0178  1     : JSB_FMTO NOVALUE ≡
:  116    0179  1
:  117    0180  1  !++
:  118    0181  1  !
:  119    0182  1  !   FUNCTIONAL DESCRIPTION:
:  120    0183  1  !
:  121    0184  1  !       Initializes the format interpreter
:  122    0185  1  !
:  123    0186  1  !   IMPLICIT INPUTS:
:  124    0187  1  !
:  125    0188  1  !       CCB                      Contains adr. of current LUB/ISB/RAB.
:  126    0189  1  !
:  127    0190  1  !   IMPLICIT OUTPUTS:
:  128    0191  1  !
:  129    0192  1  !       CCB [ISB$W_FMT_REP]      Set repeat count to 0 to indicate no repeat
:  130    0193  1  !                                for this statement yet.
:  131    0194  1  !       CCB [ISB$B_FMT_P]        Set P scale factor to 0 for this statement
:  132    0195  1  !       CCB [ISB$A_FMT_PTR]      Initializes format pointer to
:  133    0196  1  !                                beginning
:  134    0197  1  !       CCB [ISB$W_FMT_REVER]    Offset of current format reversion
:  135    0198  1  !                                point
:  136    0199  1  !       CCB [ISB$B_FMT_DEP]      Depth of repeat group pushdown stack
:  137    0200  1  !
:  138    0201  1  !   SIDE EFFECTS:
:  139    0202  1  !
:  140    0203  1  !       NONE
:  141    0204  1  !
:  142    0205  1  !--
:  143    0206  1
:  144    0207  2     BEGIN
:  145    0208  2
:  146    0209  2        EXTERNAL REGISTER
:  147    0210  2           CCB : REF $FOR$CCB_DECL;
:  148    0211  2
:  149    0212  2     !+
:  150    0213  2     ! Set repeat count to 0 to indicate no repeat for this statement.
:  151    0214  2     !-
:  152    0215  2
:  153    0216  2     CCB [ISB$W_FMT_REP] = 0;
:  154    0217  2
:  155    0218  2     !+
:  156    0219  2     ! Set P scale factor to 0 for this statement (no scaling).
:  157    0220  2     !-
:  158    0221  2
:  159    0222  2     CCB [ISB$B_FMT_P] = 0;
:  160    0223  2
:  161    0224  2     !+
:  162    0225  2     ! Set format flags to zero for this statement.
:  163    0226  2     !-
:  164    0227  2
:  165    0228  2     CCB [ISB$W_FMT_FLAGS] = 0;
:  166    0229  2
:  167    0230  2     !+
:  168    0231  2     ! Set BN flag if LUB$V_NULLBLNK is set
:  169    0232  2     !-
:  170    0233  2
```

```
 171      0234  2               CCB [ISB$V_BN] = .CCB [LUB$V_NULLBLNK];
 172      0235  2
 173      0236  2
 174      0237  2               !+
 175      0238  2               ! Set current format position to beginning of format.
 176      0239  2               !-
 177      0240  2
 178      0241  2               CCB [ISB$A_FMT_PTR] = .CCB [ISB$A_FMT_BEG];
 179      0242  2
 180      0243  2               !+
 181      0244  2               ! Initialize format reversion point to beginning of format
 182      0245  2               ! byte array. The reversion point is used when there are
 183      0246  2               ! more user data elements than data format codes.
 184      0247  2               ! Since it is a 16-bit offset with respect to ISB$A_FMT_BEG, set to 0.
 185      0248  2               !-
 186      0249  2
 187      0250  2               CCB [ISB$W_FMT_REVER] = 0;
 188      0251  2
 189      0252  2               !+
 190      0253  2               ! Initialize format repeat group push down stack depth
 191      0254  2               ! to empty (-1).  0 = 1 item, 1 = 2 items in stack, etc.
 192      0255  2               !-
 193      0256  2
 194      0257  2               CCB [ISB$B_FMT_DEP] = -1;
 195      0258  2
 196      0259  2               !+
 197      0260  2               ! Initialize ISB$B_FMT_CODE to zero, which will tell
 198      0261  2               ! FOR$$UDF_WF9 not to call FOR$$UDF_WF1 unless there were no
 199      0262  2               ! items in the I/O list.
 200      0263  2               !-
 201      0264  2
 202      0265  2               CCB [ISB$B_FMT_CODE] = 0;
 203      0266  2
 204      0267  2               !+
 205      0268  2               ! All other ISB locations and flags have already been
 206      0269  2               ! initialized to 0 or a specified value by the I/O statement
 207      0270  2               ! initialization for this I/O statement.
 208      0271  2               !-
 209      0272  2               RETURN;
 210      0273  1               END;                                    ! End of routine FOR$$FMT_INTRP0
```

```
                                        .TITLE  FOR$$FMT_INTRP Fortran Format Statement Interpr
                                                                eter
                                        .IDENT  \2-037\

                                        .EXTRN  FOR$$SIGNAL_STO
                                        .EXTRN  FOR$$SIGNAL

                                        .PSECT  _FOR$CODE,NOWRT,  SHR,  PIC,2

                          8D   AB  B4 00000 FOR$$FMT_INTRP0::
                                                CLRW    -115(CCB)                          ; 0216
                          88   AB  94 00003        CLRB    -120(CCB)                       ; 0222
                          93   AB  B4 00006        CLRW    -109(CCB)                       ; 0228
        50    FF  AB           01   06   EF 00009  EXTZV   #6, #1, -1(CCB), R0             ; 0234
   93   AB           01   00        50   F0 0000F  INSV    R0, #0, #1, -109(CCB)
```

```
                        80   AB     FF7C   CB  D0 00015          MOVL     -132(CCB), -128(CCB)              ;  0240
                                       90   AB  B4 0001B          CLRW     -112(CCB)                        ;  0549
                        92   AB             01  8E 0001E          MNEGB    #1, -110(CCB)                    ;  0256
                                       8F   AB  94 00022          CLRB     -113(CCB)                        ;  0264
                                            05 00025             RSB                                        ;  0273
```

; Routine Size:  38 bytes,     Routine Base:  _FOR$CODE + 0000

```
212   0274   1   GLOBAL ROUTINE FOR$$FMT_INTRP1              ! Format interpret until data code
213   0275   1        (EL_SIZE,                              ! Size in addressable units of data elements
214   0276   1                                               ! This argument is passed in
215   0277   1                                               ! external register EL_SIZE.
216   0278   1   !      DT_SEEN)                              ! 1 if data transmitter seen
217   0279   1                                               ! This argument is passed in
218   0280   1                                               ! external register DT_SEEN.
219   0281   1        : JSB_FMT1 NOVALUE =                   ! Value is format code
220   0282   1                                               ! Returned in external register
221   0283   1                                               ! FMT_CODE.
222   0284   1
223   0285   1   !++
224   0286   1   !
225   0287   1   !   FUNCTIONAL DESCRIPTION:
226   0288   1   !
227   0289   1   !       FOR$$FMT_INTRP1 interprets FORTRAN format statements
228   0290   1   !       which have been precompiled into an encoded form by either
229   0291   1   !       the FORTRAN compiler or the run-time format compiler,
230   0292   1   !       FOR$FMT_COMPIL. Only FOR$$FMT_INTRP1 understands the structure
231   0293   1   !       and encoding of compiled format statements. Furthermore,
232   0294   1   !       it is independent of READ and WRITE semantics.
233   0295   1   !       Each call to FOR$$FMT_INTRP1 processes as many format
234   0296   1   !       codes as possible until it encounters one which
235   0297   1   !       needs to access user program data, needs to access the
236   0298   1   !       data buffer, or depend on whether read or write. This
237   0299   1   !       block is independent of whether a READ or WRITE is
238   0300   1   !       being performed. It is invoked in both the formatted
239   0301   1   !       READ and WRITE user data formatter routines (UDF).
240   0302   1   !
241   0303   1   !       Note: being compatible with -11 OTS, there is a
242   0304   1   !       difference between nf and n(f) in that VFE's are evaluated only
243   0305   1   !       once for the former and each time for the latter. The former
244   0306   1   !       is termed, repeating a format code while the latter is
245   0307   1   !       termed a repeat group.
246   0308   1   !
247   0309   1   !   IMPLICIT INPUTS:
248   0310   1   !
249   0311   1   !       EL_SIZE                    Size in addressable units of user
250   0312   1   !                                  data element. Used to set default
251   0313   1   !                                  widths (W) for default format.
252   0314   1   !                                  A value of 0 indicates that this is
253   0315   1   !                                  the end of the I/O list call and there is
254   0316   1   !                                  no user I/O list element to be transmitted.
255   0317   1   !
256   0318   1   !       DT_SEEN                    1 if a data transmitter has been seen by
257   0319   1   !                                  the current call to the UDF level, 0
258   0320   1   !                                  otherwise. If it is set, we don't evaluate
259   0321   1   !                                  format items for data transmitters.
260   0322   1   !
261   0323   1   !       OTS$$A_CUR_LUB             Adr. of current logical unit block
262   0324   1   !                                  (LUB) Used to setup base (ISB) to
263   0325   1   !                                  I/O statement block.
264   0326   1   !
265   0327   1   !       The following locations are set only by previous calls to
266   0328   1   !       FOR$$FMT_INTRP1, i.e., are effectively OWN:
267   0329   1   !
268   0330   1   !       CCB [ISB$A_FMT_PTR]        Adr. of next byte in compiled
```

```
  269   0331  1 |                                            format byte array. A value of 0
  270   0332  1 |                                            indicates that this is the end of the
  271   0333  1 |                                            I/O list call and there is no user
  272   0334  1 |                                            I/O list element to be transmitted.
  273   0335  1 |      CCB [ISB$V_USER_ELEM]                 0 until a user element format
  274   0336  1 |                                            code seen. Infinite loop preventer
  275   0337  1 |      CCB [ISB$W_FMT_REP]                   Current format code repeat count (n)
  276   0338  1 |                                            or 0 if not repeating a single
  277   0339  1 |                                            format code. Note: the repeat
  278   0340  1 |                                            count for a repeat group is kept
  279   0341  1 |                                            in the top of the format stack, not here.
  280   0342  1 |
  281   0343  1 |      CCB [ISB$A_FMT_BEG]                   Adr. of beginning of format statement
  282   0344  1 |      CCB [ISB$B_FMT_DEP]                   Depth of repeat group format pushdown stack.
  283   0345  1 |      CCB [ISB$W_FMT_STKP]                  Stack of offsets to beginning of repeat groups
  284   0346  1 |      CCB [ISB$W_FMT_STKR]                  Stack of group repeat counts
  285   0347  1 |      CCB [ISB$W_FMT_REVER]                 Offset of current format reversion
  286   0348  1 |                                            point to revert to when end of format
  287   0349  1 |                                            statement is encountered with more data
  288   0350  1 |                                            elements to be transmitted.
  289   0351  1 |      CCB [ISB$V_USER_ELEM]                 Flag: 1 if seen a user data element format code,
  290   0352  1 |                                            0 if not.  Used to check for infinite format loop
  291   0353  1 |                                            in which no user data element format codes are present
  292   0354  1 |
  293   0355  1 |  IMPLICIT OUTPUTS:
  294   0356  1 |
  295   0357  1 |    The following are outputs only to a successive call to
  296   0358  1 |    FOR$$FMT_INTRP(0,1), i.e., are effectively OWN.
  297   0359  1 |
  298   0360  1 |      CCB [ISB$V_USER_ELEM]                 0 if no user data element format
  299   0361  1 |                                            code seen this repeat group, 1
  300   0362  1 |                                            if one or more
  301   0363  1 |      CCB [ISB$W_FMT_REP]                   Current format code repeat count (n)
  302   0364  1 |                                            or 0 if not repeating a single
  303   0365  1 |                                            format code. Note: the repeat
  304   0366  1 |                                            count for a repeat group is kept
  305   0367  1 |                                            in the top of the format stack, not here.
  306   0368  1 |      CCB [ISB$B_FMT_DEP]                   Depth of repeat group format pushdown stack.
  307   0369  1 |      CCB [ISB$W_FMT_STKP]                  Stack of offsets to beginning of repeat groups
  308   0370  1 |      CCB [ISB$W_FMT_STKR]                  Stack of group repeat counts
  309   0371  1 |      CCB [ISB$W_FMT_REVER]                 Offset of current format reversion
  310   0372  1 |                                            point to revert to when end of format
  311   0373  1 |                                            statement is encountered with more data
  312   0374  1 |                                            elements to be transmitted.
  313   0375  1 |
  314   0376  1 |    The following are output to available to the caller (read
  315   0377  1 |    or write user data formatter):
  316   0378  1 |
  317   0379  1 |      CCB [ISB$A_FMT_PTR]                   Adr. of next byte to be read from
  318   0380  1 |                                            the compiled format statement byte array
  319   0381  1 |                                            are pushed as a pair.
  320   0382  1 |      CCB [ISB$B_FMT_P]                     Signed scale factor (P)
  321   0383  1 |      CCB [ISB$W_FMT_W]                     Unsigned width of field (W)
  322   0384  1 |      CCB [ISB$B_FMT_D]                     Unsigned number of digits in fraction (D)
  323   0385  1 |      CCB [ISB$B_FMT_E]                     Unsigned number of characters
  324   0386  1 |                                            in exponent (E).
  325   0387  1 |      CCB [ISB$V_USER_ELEM]                 Flag: 1 if seen a user data element format code,
```

```
: 326      0388  1 |                             0 if not.  Used to check for infinite format loop
: 327      0389  1 |                             in which no user data element format codes are present
: 328      0390  1 |
: 329      0391  1 |  SIDE EFFECTS:
: 330      0392  1 |
: 331      0393  1 |        SIGNAL_STOPs FOR$_SYNERRFOR (62='SYNTAX ERROR IN FORMAT')
: 332      0394  1 |        SIGNAL_STOPs FOR$_INFFORLOP (60="INFINITE FORMAT LOOP")
: 333      0395  1 |        SIGNAL_STOPs FOR$_VFEVALERR (68='VFE VALUE ERROR')
: 334      0396  1 |--
```

```
336        0397  1 !
337        0398  2        BEGIN
338        0399  2
339        0400  2        EXTERNAL REGISTER
340        0401  2            CCB : REF $FOR$CCB_DECL,          ! Pointer to Common Control Block
341        0402  2            EL_SIZE,                          ! Element size (1st argument)
342        0403  2            DT_SEEN,                          ! Data transmitter seen (2nd argument)
343        0404  2            FMT_CODE : BLOCK [1, LONG];       ! Format code (return value)
344        0405  2
345        0406  2        BUILTIN
346        0407  2            TESTBITSC;
347        0408  2
348        0409  2        MACRO                                 ! Field definitions for action table
349        0410  2            FI_STOP = 0,6,1,0 %,              ! Stop if DT_SEEN
350        0411  2            FI_GETW = 0,5,1,0 %,              ! Get w value for format
351        0412  2            FI_GETD = 0,4,1,0 %,              ! Get d value for format
352        0413  2            FI_GETE = 0,3,1,0 %,              ! Get e value for format
353        0414  2            FI_USER = 0,2,1,0 %,              ! Format code involves user data element
354        0415  2            FI_EXIT = 0,1,1,0 %,              ! Exit from format interpreter loop
355        0416  2            FI_ACTION = 0,0,1,0 %;            ! Code-specific action required;
356        0417  2                                             !  see FI_ACT_2 for action
357        0418  2
358        0419  2                                             ! MAINTENANCE NOTE: An optimization
359        0420  2                                             ! below assumes knowledge of certain
360        0421  2                                             ! bit settings in FI_ACT.  See comments.
361        0422  2
362        0423  2        MACRO
363        0424  2            FI_ALL =                          ! Enumerate all format codes
364   M    0425  2 !                W D E U E S
365   M    0426  2 !                  S X T
366   M    0427  2 !                  E I O
367   M    0428  2 !                  R T P
368   M    0429  2            FI_PACK(0,0,0,0,1,1, 0),  ! ER   = 0,   ! 00  ! Format syntax error
369   M    0430  2            FI_PACK(0,0,0,0,0,0, 2),  ! LP   = 1,   ! 01  ! ( - Format reversin point
370   M    0431  2            FI_PACK(0,0,0,0,0,0, 3),  ! NLP  = 2,   ! 02  ! n( - Left paren of repeat group
371   M    0432  2            FI_PACK(0,0,0,0,0,0, 4),  ! RP   = 3,   ! 03  ! ) - Right paren of repeat group
372   M    0433  2            FI_PACK(0,0,0,0,1,1, 5),  ! EOF  = 4,   ! 04  ! ) - End of format
373   M    0434  2            FI_PACK(0,0,0,0,1,1, 1),  ! SLS  = 5,   ! 05  ! / - Record separator
374   M    0435  2            FI_PACK(0,0,0,0,1,0, 1),  ! DLR  = 6,   ! 06  ! $ - Dollar sign: terminal I/O
375   M    0436  2            FI_PACK(0,0,0,0,1,1, 1),  ! CLN  = 7,   ! 07  ! : - Colon: terminate if end of list
376   M    0437  2                           0,        !     ! UNUSED 8
377   M    0438  2            FI_PACK(0,0,0,0,0,0, 7),  ! S    = 9,   ! 09  ! S - Make + optional
378   M    0439  2            FI_PACK(0,0,0,0,0,0, 8),  ! SP   = 10,  ! 0A  ! SP - force optional +
379   M    0440  2            FI_PACK(0,0,0,0,0,0, 7),  ! SS   = 11,  ! 0B  ! SS - Leave out optional +
380   M    0441  2            FI_PACK(1,0,0,0,0,0, 6),  ! -P   = 12,  ! 0C  ! sP - signed scale factor
381   M    0442  2            FI_PACK(1,0,0,0,0,0,11),  ! -T   = 13,  ! 0D  ! Tn - Tab Set
382   M    0443  2            FI_PACK(1,0,0,0,0,0, 1),  ! -X   = 14,  ! 0E  ! old nX
383   M    0444  2            FI_PACK(0,0,0,1,0,0, 1),  ! -H   = 15,  ! 0F  ! nHcccc - Hollerith
384   M    0445  2            FI_PACK(0,0,0,0,0,0, 9),  ! BN   = 16,  ! 10  ! BN = Blanks are nulls
385   M    0446  2            FI_PACK(0,0,0,0,0,0,10),  ! BZ   = 17,  ! 11  ! BZ = Blanks are zeroes
386   M    0447  2            FI_PACK(1,0,0,0,0,0,12),  ! TL   = 18,  ! 12  ! TLn = Tab left n columns
387   M    0448  2            FI_PACK(1,0,0,0,0,0,13),  ! TR   = 19,  ! 13  ! TRn (new nX) = Tab right n columns
388   M    0449  2            FI_PACK(0,0,0,1,1,1, 1),  ! Q    = 20,  ! 14  ! Q
389   M    0450  2            FI_PACK(1,0,0,1,1,1, 1),  ! -A   = 21,  ! 15  ! nAw - Alpha numeric
390   M    0451  2            FI_PACK(1,0,0,1,1,1, 1),  ! -L   = 22,  ! 16  ! nLw - Logical
391   M    0452  2            FI_PACK(1,0,0,1,1,1, 1),  ! -O   = 23,  ! 17  ! nOw - Octal
392   M    0453  2            FI_PACK(1,0,0,1,1,1, 1),  ! -I   = 24,  ! 18  ! nIw - Integer
```

```
M 13
 393      M 0454   2              FI_PACK(1,0,0,1,1,1, 1),      !  Z    = 25,  ! 19   ! nZw - Hexadecimal
 394      M 0455   2              FI_PACK(1,1,0,1,1,1, 1),      !  XO   = 26,  ! 1A   ! Ow.m Octal
 395      M 0456   2              FI_PACK(1,1,0,1,1,1, 1),      !  XI   = 27,  ! 1B   ! Iw.m Integer
 396      M 0457   2              FI_PACK(1,1,0,1,1,1, 1),      !  XZ   = 28   ! 1C   ! Zw.m Hexadecimal
 397      M 0458   2                     0,                    ! UNUSED       29
 398      M 0459   2              FI_PACK(1,1,0,1,1,1, 1),      !  _F   = 30,  ! 1E   ! nFw.d - Fixed format
 399      M 0460   2              FI_PACK(1,1,0,1,1,1, 1),      !  _E   = 31,  ! 1F   ! nEw.d - Scientific notation format
 400      M 0461   2              FI_PACK(1,1,0,1,1,1, 1),      !  _G   = 32,  ! 20   ! nGw.d - General format
 401      M 0462   2              FI_PACK(1,1,0,1,1,1, 1),      !  _D   = 33,  ! 21   ! nDw.d - Double Precision format
 402      M 0463   2              FI_PACK(1,1,1,1,1,1, 1),      !  XE   = 34,  ! 22   ! nEw.dEe
 403      M 0464   2              FI_PACK(1,1,1,1,1,1, 1),      !  XG   = 35,  ! 23   ! nGw.dEe
 404      M 0465   2                     0,0,0,0,0,            ! UNUSED   36:40
 405      M 0466   2              FI_PACK(0,0,0,1,1,1, 1),      !  _DA  = 41,  ! 29   ! nA - default A
 406      M 0467   2              FI_PACK(0,0,0,1,1,1, 1),      !  _DL  = 42,  ! 2A   ! nL - default L
 407      M 0468   2              FI_PACK(0,0,0,1,1,1, 1),      !  _DO  = 43,  ! 2B   ! nO - default O
 408      M 0469   2              FI_PACK(0,0,0,1,1,1, 1),      !  _DI  = 44,  ! 2C   ! nI - default I
 409      M 0470   2              FI_PACK(0,0,0,1,1,1, 1),      !  _DZ  = 45,  ! 2D   ! nZ - default Z
 410      M 0471   2                     0,0,0,0,              ! UNUSED   46:49
 411      M 0472   2              FI_PACK(0,0,0,1,1,1, 1),      !  _DF  = 50,  ! 32   ! nF - default F
 412      M 0473   2              FI_PACK(0,0,0,1,1,1, 1),      !  _DE  = 51,  ! 33   ! nE - default E
 413      M 0474   2              FI_PACK(0,0,0,1,1,1, 1),      !  _DG  = 52,  ! 34   ! nG - default G
 414      M 0475   2              FI_PACK(0,0,0,1,1,1, 1)       !  _DD  = 53,  ! 35   ! nD - default D
 415        0476   2         X;                                ! End of macro FI_ALL
 416        0477   2
 417        0478   2     !+
 418        0479   2     ! Define FI_PACK for use in constructing FI_ACT
 419        0480   2     !-
 420        0481   2
 421        0482   2         MACRO                                 ! Attributes-packing macro for attributes table
 422      M 0483   2             FI_PACK (W, D, E, U, X, S, NDX) =
 423      M 0484   2                 (S^6 + W^5 + D^4 + E^3 + U^2 + X^1 +
 424        0485   2                  %IF %IDENTICAL (NDX, 1) %THEN 0 %ELSE 1 %FI) %;
 425        0486   2
 426        0487   2         BIND
 427        0488   2             FI_ACT =                          ! First action table
 428        0489   2                 UPLIT BYTE ( FI_ALL ) : VECTOR [54, BYTE];
 429        0490   2
 430        0491   2     !+
 431        0492   2     ! Redefine FI_PACK for use in constructing FI_ACT_2
 432        0493   2     !-
 433        0494   2
 434        0495   2         UNDECLARE %QUOTE FI_PACK;
 435        0496   2
 436        0497   2         MACRO
 437      M 0498   2             FI_PACK (W, D, E, U, X, S, NDX) =
 438        0499   2                 NDX %;
 439        0500   2
 440        0501   2         BIND
 441        0502   2             FI_ACT_2 =                        ! Second action table
 442        0503   2                 UPLIT BYTE ( FI_ALL ) : VECTOR [54, BYTE];
 443        0504   2
 444        0505   2 !<BLF/PAGE>
```

```
 446    0506   2            !+
 447    0507   2            ! (NXTITM+1)
 448    0508   2            ! Assume that a format code is being repeated.- nf not n(f).
 449    0509   2            ! (as distinguished from a repeat group which is n(...))
 450    0510   2            ! Decrement format repeat count (ISB$W_FMT_REP). Test
 451    0511   2            ! if still more to repeat - if yes, skip usual format code
 452    0512   2            ! dispatching by skipping loop altogether, redo defaults if
 453    0513   2            ! default format codes and RETURN
 454    0514   2            !-
 455    0515   2
 456    0516   2            IF .CCB [ISB$W_FMT_REP] GTR 1
 457    0517   2            THEN
 458    0518   2                BEGIN
 459    0519   2
 460    0520   2                LOCAL
 461    0521   3                    ACT : BLOCK [1, LONG];        ! Action table entry for format code
 462    0522   3
 463    0523   3                FMT_CODE = .CCB [ISB$B_FMT_CODE];
 464    0524   3                ACT = .FI_ACT [.FMT_CODE];
 465    0525   3                IF .DT_SEEN AND .ACT [FI_STOP]
 466    0526   3                THEN
 467    0527   3                    BEGIN
 468    0528   4                    FMT_CODE = 0;
 469    0529   4                    RETURN;
 470    0530   4                    END;
 471    0531   3                CCB [ISB$W_FMT_REP] = .CCB [ISB$W_FMT_REP] - 1;
 472    0532   3                END
 473    0533   2            ELSE
 474    0534   2
 475    0535   2            !+
 476    0536   2            ! (FINTRP)
 477    0537   2            ! Not in format code repeat - start format interpret loop
 478    0538   2            ! Loop until encounter a format code which needs to access
 479    0539   2            ! data (ER or explicit or default Q, A, L, O, I, Z, F, E, G, or D),
 480    0540   2            ! needs to access the data buffer (X, H, Q), or
 481    0541   2            ! depends on whether read or write (), /, $, :, T).
 482    0542   2            !-
 483    0543   2
 484    0544   2                BEGIN
 485    0545   2
 486    0546   2                REGISTER
 487    0547   3                    P,                            ! Pointer to format byte stream
 488    0548   3                    ACT : BLOCK [1, LONG];        ! Action table entry for format code
 489    0549   3
 490    0550   3                P = .CCB [ISB$A_FMT_PTR];
 491    0551   3
 492    0552   3                DO
 493    0553   3                    BEGIN
 494    0554   4
 495    0555   4                    !+
 496    0556   4                    ! Pickup next format code byte from compiled format:
 497    0557   4                    ! If optional representation byte
 498    0558   4                    ! is present (V_FMT_REPRE=1), mask out flag bit
 499    0559   4                    ! in format code and copy next byte to BITVECTOR
 500    0560   4                    ! to indicate larger (less frequent) sizes of the
 501    0561   4                    ! code representation or Variable Field Expressions (VFE).
 502    0562   4
```

```
503  0563  4                      !-
504  0564  4
505  0565  4                      FMT_CODE = CH$RCHAR (.P);
506  0566  4                      FMT_CODE [V_FMT_REPRE] = 0;           ! Clear bit for search
507  0567  4                      ACT = .FI_ACT [.FMT_CODE];
508  0568  4
509  0569  4                      !+
510  0570  4                      ! If DT_SEEN is set and this format code needs a data transmitter
511  0571  4                      ! then return a format code of
512  0572  4                      ! zero to signal the fact.  This will be differentiated from
513  0573  4                      ! an error by the UDF level by checking DT_SEEN.
514  0574  4                      !-
515  0575  4
516  0576  4                      IF .DT_SEEN AND .ACT [FI_STOP]
517  0577  4                      THEN
518  0578  5                          BEGIN
519  0579  5                          CCB [ISB$A_FMT_PTR] = .P;
520  0580  5                          FMT_CODE = 0;
521  0581  5                          RETURN;
522  0582  4                          END;
523  0583  4
524  0584  4                      FMT_CODE = CH$RCHAR_A (P);            ! Re-read and increment pointer
525  0585  4
526  0586  4                      !+
527  0587  4                      ! Optimization:
528  0588  4                      !
529  0589  4                      ! Check for certain easily-handled (and frequent) cases:
530  0590  4                      ! 1.  A/L/O/I/Z (codes 21-25) with no RC and byte-length W;
531  0591  4                      ! 2.  O/I/Z/F/E/D/G (codes 26-28 and 30-33) with no RC and
532  0592  4                      !     byte-length W, D;
533  0593  4                      ! 3.  E/G (codes 34-35) with no RC and byte-length W, D, E;
534  0594  4                      ! If found, handle directly and bypass the tests for VFE's,
535  0595  4                      ! word-length RC and W, and special action.  Note that
536  0596  4                      ! anything with V_FMT_REPRE set falls under OUTRANGE.
537  0597  4                      !
538  0598  4                      ! This optimization assumes knowledge of flag bit settings
539  0599  4                      ! in FI_ACT, and must be reconsidered if FI_ACT changes.
540  0600  4                      !-
541  0601  4
542  0602  5                      IF NOT (CASE .FMT_CODE FROM _A TO XG OF
543  0603  5                          SET
544  0604  5                              [_A TO _Z] :
545  0605  6                                  BEGIN
546  0606  6                                  CCB [ISB$W_FMT_W] = RBYTE_A (P);
547  0607  6                                  CCB [ISB$W_FMT_REP] = 1;
548  0608  6                                  CCB [ISB$V_USER_ELEM] = 1;
549  0609  6                                  1                    ! Indicate special case found
550  0610  5                                  END;
551  0611  5                              [XO TO XZ, _F TO _D] :
552  0612  6                                  BEGIN
553  0613  6                                  CCB [ISB$W_FMT_W] = RBYTE_A (P);
554  0614  6                                  CCB [ISB$B_FMT_D] = RBYTE_A (P);
555  0615  6                                  CCB [ISB$B_FMT_E] = 2;
556  0616  6                                  CCB [ISB$W_FMT_REP] = 1;
557  0617  6                                  CCB [ISB$V_USER_ELEM] = 1;
558  0618  6                                  1                    ! Indicate special case found
559  0619  5                                  END;
```

```
560   0620  5                          [XE TO XG] :
561   0621  6                              BEGIN
562   0622  6                              CCB [ISBSW_FMT_W] = RBYTE_A (P);
563   0623  6                              CCB [ISBSB_FMT_D] = RBYTE_A (P);
564   0624  6                              CCB [ISBSB_FMT_E] = RBYTE_A (P);
565   0625  6                              CCB [ISBSW_FMT_REP] = 1;
566   0626  6                              CCB [ISBSV_USER_ELEM] = 1;
567   0627  6                              1                  ! Indicate special case found
568   0628  6                              END;
569   0629  5                          [29, OUTRANGE] :
570   0630  5                              0;                 ! Indicate special case not found
571   0631  5                          TES)
572   0632  4                  THEN
573   0633  5                      BEGIN
574   0634  5
575   0635  5                      !+
576   0636  5                      ! Get RC, W, D and E in the traditional, fully general
577   0637  5                      ! way, including check for VFE's and alternate forms of
578   0638  5                      ! W and RC.
579   0639  5                      !
580   0640  5                      ! Optimization:
581   0641  5                      !
582   0642  5                      ! If optional second byte is not present, bypass check
583   0643  5                      ! for VFE's and alternate forms of W and RC.
584   0644  5                      !-
585   0645  5
586   0646  5                      IF NOT TESTBITSC (FMT_CODE [V_FMT_REPRE])
587   0647  5                      THEN
588   0648  6                          BEGIN                       ! Begin short form
589   0649  6                          CCB [ISBSW_FMT_REP] = 1;
590   0650  6                          IF .ACT [FI_GETW]
591   0651  6                          THEN
592   0652  7                              BEGIN
593   0653  7                              CCB [ISBSW_FMT_W] = RBYTE_A (P);
594   0654  7                              IF .ACT [FI_GETD]
595   0655  7                              THEN
596   0656  8                                  BEGIN
597   0657  8                                  CCB [ISBSB_FMT_D] = RBYTE_A (P);
598   0658  8                                  IF .ACT [FI_GETE]
599   0659  8                                  THEN
600   0660  8                                      CCB [ISBSB_FMT_E] = RBYTE_A (P);
601   0661  7                                  END;
602   0662  6                              END;
603   0663  6                          END                         ! End short form
604   0664  5                      ELSE
605   0665  6                          BEGIN                       ! Begin long form
606   0666  6
607   0667  6                          LOCAL
608   0668  6                              FMT_REPR : BLOCK [1, LONG];
609   0669  6
610   0670  6                          FMT_REPR = RBYTE_A (P);
611   0671  6
612   0672  6                          !+
613   0673  6                          ! Get repeat count (RC) from format and save in ISBSW_FMT_REP.
614   0674  6                          ! If repeat count is a VFE (FMT_REPR[V_RC_VFE]=1), get VFE and
615   0675  6                          ! check for out of range (1:32767).
616   0676  6                          ! If explicitly represented, get byte or word value.
```

```
617    0677    6        ! Else set repeat count to 1. Possible for left paren
618    0678    6        ! of a repeat group (NLP) or A, L, O, Z, I, F, E, G, D
619    0679    6        ! or default A, L, O, Z, I, F, E, G, D.
620    0680    6        !_
621    0681    6
622    0682    7        CCB [ISBSW_FMT_REP] = (IF .FMT_REPR [V_RC_VFE]
623    0683    7            THEN
624    0684    8            BEGIN                    ! Process RC VFE
625    0685    8
626    0686    8            LOCAL
627    0687    8                T;
628    0688    8
629    0689    8            T = CALL_VFE (P);
630    0690    8
631    0691    8            IF .T GEQU 32768 OR .T EQL 0
632    0692    8            THEN
633    0693    9                BEGIN
634    0694    9                FOR$$SIGNAL (FOR$K_VFEVALERR);
635    0695    9                1                    ! Force repeat count to 1 on error
636    0696    9                END
637    0697    8            ELSE
638    0698    8                .T                   ! Use user supplied value
639    0699    8            END                      ! End of RC VFE processing
640    0700    8
641    0701    7        ELSE
642    0702    7
643    0703    7            ! The following assumes that RC is either 00
644    0704    7            ! (absent), 01 (byte) or 10 (word), and that
645    0705    7            ! it cannot be 11.
646    0706    7
647    0707    7            IF .FMT_REPR [V_RC_TYPE_BYTE]
648    0708    7            THEN
649    0709    8                RBYTE_A (P)          ! RC is a byte
650    0710    7            ELSE
651    0711    7                IF .FMT_REPR [V_RC_TYPE_WORD]
652    0712    7                THEN
653    0713    8                    RWORD_A (P)      ! RC is a word
654    0714    7                ELSE
655    0715    6                    1);              ! RC is absent
656    0716    6
657    0717    6        !+
658    0718    6        ! P, T, X, H, A ,L, O, I, Z, F, E, G, D:
659    0719    6        ! Get field width (w) from format and
660    0720    6        ! set ISBSW_FMT_W. If width field is a
661    0721    6        ! VFE (V_W_VFE=T), get VFE value and check range;
662    0722    6        ! if P scale -128 to 127, else (field width w) 0 to 32767.
663    0723    6        ! If width of field is a byte (V_W_WORD=0), get byte
664    0724    6        ! else get word.  ISBSW_FMT_W is set as a
665    0725    6        ! zero extended word.
666    0726    6        !_
667    0727    6
668    0728    6        IF .ACT [FI_GETW]
669    0729    6        THEN
670    0730    7            BEGIN
671    0731    8            CCB [ISBSW_FMT_W] = (IF .FMT_REPR [V_W_VFE] THEN
672    0732    9                BEGIN
673    0733    9
```

```
  674    0734  9                                      LOCAL
  675    0735  9                                          T;
  676    0736  9
  677    0737  9                                      T = CALL_VFE (P);
  678    0738  9
  679    0739  9                                      IF .FMT_CODE EQL _P
  680    0740  9                                      THEN
  681    0741 10                                          BEGIN                     ! P scale
  682    0742 10
  683    0743 10                                          IF .T<0,8,1> NEQ .T       ! P between -128 and 127?
  684    0744 10                                          THEN
  685    0745 11                                              BEGIN
  686    0746 11                                              FOR$$SIGNAL (FOR$K_VFEVALERR);
  687    0747 11                                              T = 0                     ! Force P scale to 0
  688    0748 11                                              END
  689    0749 11
  690    0750 10                                          END
  691    0751  9                                      ELSE
  692    0752 10                                          BEGIN                             ! Else w width of field
  693    0753 10
  694    0754 10                                          IF .T GEQU 32768
  695    0755 10                                          THEN
  696    0756 11                                              BEGIN
  697    0757 11                                              FOR$$SIGNAL (FOR$K_VFEVALERR);
  698    0758 11                                              T = 1
  699    0759 11                                              END
  700    0760 11
  701    0761  9                                          END;
  702    0762  9
  703    0763  9                                      .T                        ! return VFE value
  704    0764  9                                      END
  705    0765  7                                  ELSE IF .FMT_REPR [V_W_WORD] THEN RWORD_A (P) ELSE RBYTE_A (P));
  706    0766  7
  707    0767  7                                  !+
  708    0768  7                                  ! Get decimal part (d) from format and set
  709    0769  7                                  ! ISBSB_FMT_D. If decimal part is a VFE
  710    0770  7                                  ! (V_D_VFE=T) get VFE and check range (0:32767).
  711    0771  7                                  ! Else get byte from format
  712    0772  7                                  !
  713    0773  7                                  ! Set default exponent width to 2.
  714    0774  7                                  !-
  715    0775  7
  716    0776  7                                  IF .ACT [FI_GETD]
  717    0777  7                                  THEN
  718    0778  8                                      BEGIN
  719    0779  9                                      CCB [ISBSB_FMT_D] = (IF .FMT_REPR [V_D_VFE] THEN
  720    0780 10                                          BEGIN                     ! VFE
  721    0781 10
  722    0782 10                                          LOCAL
  723    0783 10                                              T;
  724    0784 10
  725    0785 10                                          T = CALL_VFE (P);
  726    0786 10
  727    0787 10                                          IF .T GEQU 32768
  728    0788 10                                          THEN
  729    0789 11                                              BEGIN
  730    0790 11                                              FOR$$SIGNAL (FOR$K_VFEVALERR);
```

```
731    0791 11                                            1
732    0792 11                                          END
733    0793 10                                  ELSE
734    0794 10                                    .T
735    0795 10
736    0796 10                                  END
737    0797  8                          ELSE RBYTE_A (P));
738    0798  8                          CCB [ISB$B_FMT_E] = 2;
739    0799  8
740    0800  8                          !+
741    0801  8                          ! Get exponent width (e) from format and set
742    0802  8                          ! ISB$B_FMT_E.  If exponent width is a VFE, check
743    0803  8                          ! range (0:255).  Else get byte from format.
744    0804  8                          !-
745    0805  8
746    0806  8                          IF .ACT [FI_GETE]
747    0807  8                          THEN
748    0808  9                              BEGIN
749    0809 10                              CCB [ISB$B_FMT_E] = (IF .FMT_REPR [V_E_VFE] THEN
750    0810 11                                  BEGIN                         ! VFE
751    0811 11
752    0812 11                                  LOCAL
753    0813 11                                      T;
754    0814 11
755    0815 11                                  T = CALL_VFE (P);
756    0816 11
757    0817 11                                  IF .T GEQU 256
758    0818 11                                  THEN
759    0819 12                                      BEGIN
760    0820 12                                      FOR$$SIGNAL (FOR$K_VFEVALERR);
761    0821 12                                      1
762    0822 12                                      END
763    0823 11                                  ELSE
764    0824 11                                      .T
765    0825 11
766    0826 11                                  END
767    0827  9                              ELSE RBYTE_A (P));
768    0828  8                              END;
769    0829  8
770    0830  7                          END;
771    0831  7
772    0832  6                      END;
773    0833  6
774    0834  5                  END;                          ! End long form
775    0835  5
776    0836  5          !+
777    0837  5          ! For all user data element format codes
778    0838  5          ! (explicit and default Q, A, L, O, I, Z, F, E, G, D):
779    0839  5          ! Set user data element format code
780    0840  5          ! seen in this group, because not in an
781    0841  5          ! infinite format loop invoking for a user
782    0842  5          ! data element format code which doesn't exist.
783    0843  5          !-
784    0844  5
785    0845  5          IF .ACT [FI_USER] THEN CCB [ISB$V_USER_ELEM] = 1;
786    0846  5
787    0847  5          !+
```

FOR$$FMT_INTRP  Fortran Format Statement Interpreter
2-037

G 14
16-Sep-1984 00:25:18    VAX-11 Bliss-32 V4.0-742
14-Sep-1984 12:32:00    [FORRTL.SRC]FORFMTINT.B32;1

Page 18
(6)

```
788  0848  5                          ! Dispatch on format code and select appropriate actions:
789  0849  5                          !_
790  0850  5
791  0851  5                          IF .ACT [FI_ACTION]
792  0852  5                          THEN
793  0853  5                              CASE .FI_ACT_2 [.FMT_CODE] FROM 0 TO 13 OF
794  0854  5                              SET
795  0855  5
796  0856  5                              [0] :
797  0857  5
798  0858  5                                  !+
799  0859  5                                  ! ER or undefined format code
800  0860  5                                  ! Bad format: Signal_stop SYNTAX ERROR IN FORMAT (FOR$_SYNERRFOR)
801  0861  5                                  !_
802  0862  5
803  0863  6                                  BEGIN
804  0864  6                                  FOR$$SIGNAL_STO (FOR$K_SYNERRFOR);
805  0865  6                                  FMT_CODE = 0;
806  0866  6                                  RETURN;
807  0867  6                                  END;
808  0868  5
809  0869  5                              [1] :
810  0870  5
811  0871  5                                  !+
812  0872  5                                  ! No special actions required.
813  0873  5                                  !_
814  0874  5
815  0875  5
816  0876  5                                  :
817  0877  5
818  0878  5                              [2] :
819  0879  5
820  0880  5                                  !+
821  0881  5                                  ! LP  Format reversion point: left paren of
822  0882  5                                  ! second outer-most pair. Remeber current format
823  0883  5                                  ! offset (ISB$W_FMT_REVER) in case more data
824  0884  5                                  ! element in I/O list than data format
825  0885  5                                  ! codes in format. Reset push down stack to
826  0886  5                                  ! empty (-1) since this is start of
827  0887  5                                  ! first group repeat. Clear user data element
828  0888  5                                  ! seen flag (ISB$V_USER_ELEM) as a defense
829  0889  5                                  ! against infinite loop with no data
830  0890  5                                  ! transmit format code
831  0891  5                                  ! Note: format text pointer already advanced to next byte
832  0892  5                                  !_
833  0893  6                                  BEGIN
834  0894  6                                  CCB [ISB$B_FMT_DEP] = -1;
835  0895  6                                  CCB [ISB$W_FMT_REVER] = .P - .CCB [ISB$A_FMT_BEG];
836  0896  6                                  CCB [ISB$V_USER_ELEM] = 0;
837  0897  5                                  END;                    ! End LP
838  0898  5
839  0899  5                              [3] :
840  0900  5
841  0901  5                                  !+
842  0902  5                                  ! NLP   Left paren of a repeat group: Push repeat
843  0903  5                                  ! count (ISB$W_FMT_REP) and current (ISB$A_FMT_PTR)
844  0904  5                                  ! onto format stacks
```

```
 845   0905  5                                              !-
 846   0906  5
 847   0907  6                                              BEGIN
 848   0908  6                                              CCB [ISB$B_FMT_DEP] = .CCB [ISB$B_FMT_DEP] + 1;
 849   0909  6                                              VECTOR [CCB [ISB$W_FMT_STKR], .CCB [ISB$B_FMT_DEP];, WORD, UNSIGNED]     !
 850   0910  6                                              = .CCB [ISB$W_FMT_REP];
 851   0911  6                                              VECTOR [CCB [ISB$Q_FMT_STKP], .CCB [ISB$B_FMT_DEP];, WORD, UNSIGNED]     !
 852   0912  6                                              = .P - .CCB [ISB$A_FMT_BEG];
 853   0913  6                                              CCB [ISB$W_FMT_REP] = T;
 854   0914  5                                              END;                        ! End NLP
 855   0915  5
 856   0916  5                              [4] :
 857   0917  5
 858   0918  5                                              !+
 859   0919  5                                              ! RP    Right paren of repeat group: Decrement
 860   0920  5                                              ! current group repeat count (on top of
 861   0921  5                                              ! stack)  If current group count still greater
 862   0922  5                                              ! than 0, set current format pointer back to
 863   0923  5                                              ! beginning of repeat group. Else pop off
 864   0924  5                                              ! beginning of group pointer and group repeat count
 865   0925  5                                              !-
 866   0926  5
 867   0927  6                                              IF (VECTOR [CCB [ISB$W_FMT_STKR], .CCB [ISB$B_FMT_DEP];, WORD, UNSIGNED]     !
 868   0928  5                                              = .VECTOR [CCB [ISB$W_FMT_STKR], .CCB [ISB$B_FMT_DEP];, WORD, UNSIGNED] - 1) GTR
 869   0929  5                                              THEN
 870   0930  5                                              ! reset pointer to address of repeat group
 871   0931  5                                              !
 872   0932  5                                                  P = .CCB [ISB$A_FMT_BEG]      !
 873   0933  5                                                  + .VECTOR [CCB [ISB$W_FMT_STKP], .CCB [ISB$B_FMT_DEP];, WORD, UNSIGNED]
 874   0934  5                                              ELSE
 875   0935  5                                              ! pop off pointer and repeat count
 876   0936  5                                              !
 877   0937  5                                                  CCB [ISB$B_FMT_DEP] = .CCB [ISB$B_FMT_DEP] - 1;
 878   0938  5
 879   0939  5                              [5] :
 880   0940  5
 881   0941  5                                              !+
 882   0942  5                                              ! EOF    End of format:
 883   0943  5                                              ! If not end of user I/O list (EL_SIZE=0)
 884   0944  5                                              ! and no user data element format code
 885   0945  5                                              ! (ISB$V_USER_ELEM=0), then Signal_stop. INFINITE
 886   0946  5                                              ! FORMAT LOOP (FOR$_INFFORLOP).
 887   0947  5                                              ! Reset current format pointer to reversion point
 888   0948  5                                              ! (ISB$W_FMT_REVER). Initialize format stack depth.
 889   0949  5                                              !-
 890   0950  5
 891   0951  6                                              BEGIN
 892   0952  6                                              P = .CCB [ISB$A_FMT_BEG] + .CCB [ISB$W_FMT_REVER];
 893   0953  6                                              CCB [ISB$B_FMT_DEP] = -1;
 894   0954  6
 895   0955  6                                              IF .EL_SIZE GTRU 0 AND NOT .CCB [ISB$V_USER_ELEM]
 896   0956  6                                              THEN
 897   0957  7                                                  BEGIN
 898   0958  7                                                  FOR$$SIGNAL_STO (FOR$K_INFFORLOO);
 899   0959  7                                                  FMT_CODE = 0;
 900   0960  7                                                  RETURN;
 901   0961  6                                                  END;
```

```
 902   0962  6                                            END;
 903   0963  5
 904   0964  5                                    [6] :
 905   0965  5
 906   0966  5                                        !+
 907   0967  5                                        ! P    Scale factor (sP): -128 =< s =< 127
 908   0968  5                                        ! Convert unsigned word width (w) (ISB$W_FMT_W)
 909   0969  5                                        ! to signed byte ('s) and save in ISB$B_FMT_P.
 910   0970  5                                        !-
 911   0971  5
 912   0972  5                                        BEGIN
 913   0973  6                                        CCB [ISB$B_FMT_P] = .CCB [ISB$W_FMT_W];
 914   0974  6                                        END;
 915   0975  5
 916   0976  5                                    [7] :
 917   0977  5
 918   0978  5                                        !+
 919   0979  5                                        ! S, SS   Restore option of + to processor.
 920   0980  5                                        !-
 921   0981  5
 922   0982  5                                        BEGIN
 923   0983  6                                        CCB [ISB$V_SP] = 0;
 924   0984  6                                        END;
 925   0985  5
 926   0986  5                                    [8] :
 927   0987  5
 928   0988  5                                        !+
 929   0989  5                                        ! SP    Force optional + to appear
 930   0990  5                                        !-
 931   0991  5
 932   0992  5                                        BEGIN
 933   0993  6                                        CCB [ISB$V_SP] = 1;
 934   0994  6                                        END;
 935   0995  5
 936   0996  5                                    [9] :
 937   0997  5
 938   0998  5                                        !+
 939   0999  5                                        ! BN    Treat blanks as nulls on numeric input.
 940   1000  5                                        !-
 941   1001  5
 942   1002  5                                        BEGIN
 943   1003  6                                        CCB [ISB$V_BN] = 1;
 944   1004  6                                        END;
 945   1005  5
 946   1006  5                                    [10] :
 947   1007  5
 948   1008  5                                        !+
 949   1009  5                                        ! BZ    Treat blanks as zeroes on numeric input.
 950   1010  5                                        !-
 951   1011  5
 952   1012  5                                        BEGIN
 953   1013  6                                        CCB [ISB$V_BN] = 0;
 954   1014  6                                        END;
 955   1015  5
 956   1016  5                                    [11] :
 957   1017  5
 958   1018  5
```

```
959     1019  5
960     1020  5
961     1021  5                                    !+
962     1022  5                                    | Tn   Move buffer pointer to position n
963     1023  5                                    !_
964     1024  6
965     1025  6                                    BEGIN
966     1026  6                                    CCB [LUB$A_BUF_PTR] = .CCB [LUB$A_BUF_BEG] + (.CCB [ISB$W_FMT_W] - 1);
967     1027                                       END;
968     1028                              [12] :
969     1029
970     1030                                       !+
971     1031                                       | TLn  Move buffer pointer left n positions
972     1032  5                                    !_
973     1033  5
974     1034  6                                    BEGIN
975     1035  6                                    CCB [LUB$A_BUF_PTR] = .CCB [LUB$A_BUF_PTR] - .CCB [ISB$W_FMT_W];
976     1036  6
977     1037  6                                    IF .CCB [LUB$A_BUF_PTR] LSSA .CCB [LUB$A_BUF_BEG]
978     1038  6                                    THEN
979     1039  6                                        CCB [LUB$A_BUF_PTR] = .CCB [LUB$A_BUF_BEG];
980     1040  6
981     1041                                       END;
982     1042                              [13] :
983     1043
984     1044
985     1045                                       !+
986     1046                                       | TRn  Move buffer pointer right n spaces.
987     1047                                       |      Note: as of VMS Release 2, the format nX
988     1048                                       |      is equivalent to TRn. The old nX code
989     1049                                       |      is no longer generated but is supported
990     1050                                       |      for compatibility.
991     1051  5                                    !_
992     1052  5
993     1053  6                                    BEGIN
994     1054  6                                    CCB [LUB$A_BUF_PTR] = .CCB [LUB$A_BUF_PTR] + .CCB [ISB$W_FMT_W];
995     1055                                       END;
996     1056                                TES;
997     1057  5
998     1058  5                        !+
999     1059  5                        | End of loop - continue if just format control
1000    1060  5                        | ( (, n(, )) or not dependent on read/write
1001    1061  5                        | and doesn't access data buffer (P)
1002    1062  5                        |
1003    1063  5                        | EXITLOOP for format codes which access user data.
1004    1064  5                        | (ER or explicit or  default A, L, O, I, Z, F, E, G or D),
1005    1065  5                        | EXITLOOP for format codes which access data
1006    1066  5                        | buffer (X, H, Q)  EXITLOOP for format codes
1007    1067  5                        | which depend on whether read or write (end
1008    1068  5                        | of format, /, S, :, T).
1009    1069  5                        |_
1010    1070  5
1011    1071  5                    END
1012    1072  4                END
1013    1073  3          UNTIL .ACT [FI_EXIT];
1014    1074
1015    1075  3          !+
```

FOR$$FMT_INTRP   Fortran Format Statement Interpreter

K 14
16-Sep-1984 00:25:18     VAX-11 Bliss-32 V4.0-742
14-Sep-1984 12:32:00     LFORRTL.SRC]FORFMTINT.B32;1

Page 22
(6)

2-037

```
1016    1076    |  | Reset format code and updated format text pointer in ISB.
1017    1077    |  |_
1018    1078    |
1019    1079    |  CCB [ISB$B_FMT_CODE] = .FMT_CODE;
1020    1080    |  CCB [ISB$A_FMT_PTR] = .P;
1021    1081    |  END;
1022    1082    |
```

```
1024    1083  2        !+
1025    1084  2        ! Default data format codes - set defaults based on size of
1026    1085  2        ! each user data element even if inside a format code repeat
1027    1086  2        ! since the size could be different for each user data element
1028    1087  2        !-
1029    1088  2
1030    1089  2        IF .FMT_CODE GEQU _DA
1031    1090  2        THEN
1032    1091          BEGIN
1033    1092
1034    1093  3        CASE .FMT_CODE FROM _DA TO _DD OF
1035    1094              SET
1036    1095
1037    1096              [_DA] :
1038    1097
1039    1098  3              !+
1040    1099  3              ! Default A: set w field (ISB$W_FMT_W) from
1041    1100  3              ! size of user data element
1042    1101  3              !-
1043    1102
1044    1103  3              CCB [ISB$W_FMT_W] = .EL_SIZE;
1045    1104
1046    1105  3          [_DL] :
1047    1106
1048    1107  3              !+
1049    1108  3              ! Default L:  set w field (ISB$W_FMT_W) to 2
1050    1109  3              !-
1051    1110
1052    1111  3              CCB [ISB$W_FMT_W] = 2;
1053    1112
1054    1113  3          [_DI] :
1055    1114
1056    1115  3              !+
1057    1116  3              ! Default I: Set w field to 7 if element is smaller than
1058    1117  3              ! 4 bytes else set it to 12.
1059    1118  3              !-
1060    1119
1061    1120  3              IF .EL_SIZE LSSU 4 THEN CCB [ISB$W_FMT_W] = 7 ELSE CCB [ISB$W_FMT_W] = 12;
1062    1121
1063    1122  3          [_DO, _DZ] :
1064    1123
1065    1124  3              !+
1066    1125  3              ! Default O, Z.  Set to the width that would allow O
1067    1126  3              ! format plus a space. \\ Note:  For compatibility with
1068    1127  3              ! previous releases, the sizes for 1, 2 and 4 bytes must
1069    1128  3              ! be 7, 7 and 12 respectively. \\
1070    1129  3              !-
1071    1130
1072    1131  3              CCB [ISB$W_FMT_W] = MAX (7, MIN (65535, (((8*.EL_SIZE)+2)/3)+1));
1073    1132          [_DF, _DE, _DG, _DD] :
1074    1133
1075    1134  3              !+
1076    1135  3              ! Default F, E, G, D: Set w and e fields as is appropriate
1077    1136  3              ! to the element size.  Note that anything that is not
1078    1137  3              ! 8 (REAL*8) or 16 (REAL*16) is assumed to be 4 (REAL*4),
1079    1138  3              ! but check for 4 first.
1080    1139  3              !-
```

```
 1081    1140   3
 1082    1141   4                          BEGIN
 1083    1142   4
 1084    1143   4                          SELECTONE .EL_SIZE OF
 1085    1144   4                              SET
 1086    1145   4
 1087    1146   4                              [4] :
 1088    1147   5                                  BEGIN
 1089    1148   5                                  CCB [ISB$B_FMT_E] = 2;
 1090    1149   5                                  CCB [ISB$W_FMT_W] = 15;
 1091    1150   5                                  CCB [ISB$B_FMT_D] = 7;
 1092    1151   4                                  END;
 1093    1152   4
 1094    1153   4                              [8] :
 1095    1154   5                                  BEGIN
 1096    1155   5                                  CCB [ISB$B_FMT_E] = 2;
 1097    1156   5                                  CCB [ISB$W_FMT_W] = 25;
 1098    1157   5                                  CCB [ISB$B_FMT_D] = 16;
 1099    1158   5                                  END;
 1100    1159   4
 1101    1160   4                              [16] :
 1102    1161   5                                  BEGIN
 1103    1162   5                                  CCB [ISB$B_FMT_E] = 3;
 1104    1163   5                                  CCB [ISB$W_FMT_W] = 42;
 1105    1164   5                                  CCB [ISB$B_FMT_D] = 33;
 1106    1165   4                                  END;
 1107    1166   4
 1108    1167   4                              [OTHERWISE] :
 1109    1168   5                                  BEGIN
 1110    1169   5                                  CCB [ISB$B_FMT_E] = 2;
 1111    1170   5                                  CCB [ISB$W_FMT_W] = 15;
 1112    1171   5                                  CCB [ISB$B_FMT_D] = 7;
 1113    1172   4                                  END;
 1114    1173   4                              TES;
 1115    1174   4
 1116    1175   3                          END;
 1117    1176   3
 1118    1177   3                      [INRANGE] :
 1119    1178   3                          ;
 1120    1179   3                      TES;
 1121    1180   3
 1122    1181   3              !+
 1123    1182   3              ! Translate default format code to corresponding explicit code.
 1124    1183   3              !-
 1125    1184   3
 1126    1185   3          FMT_CODE = .FMT_CODE - (_DA - _A);
 1127    1186   2          END;
 1128    1187   2
 1129    1188   2      !+
 1130    1189   2      ! Return to read, write User Data Formatter (UDF). If default
 1131    1190   2      ! format code, return corresponding explicit format code
 1132    1191   2      ! to UDF. Else return the actual format code
 1133    1192   2      !-
 1134    1193   2
 1135    1194   2      RETURN;
 1136    1195   2
 1137    1196   1      END;                                        ! End of routine FOR$$FMT_INTRP1
```

FORSSFMT_INTRP  Fortran Format Statement Interpreter
2-037

N 14
16-Sep-1984 00:25:18    VAX-11 Bliss-32 V4.0-742
14-Sep-1984 12:32:00    [FORRTL.SRC]FORFMTINT.B32;1

Page 25
(7)

```
22  21  21  01  01  01  00  42  02  02  43  01  01  01  43  00026  P.AAA:  .BYTE   67, 1, 1, 1, 67, 2, 2, 66, 0, 1, 1, 1, -
00  76  76  76  66  66  66  66  66  46  21  21  01  01  22  00035                  33, 33, 34, 34, 1, 1, 33, 33, 70, 102, -
46  46  46  46  00  C0  00  00  00  7E  7E  76  76  76  76  00044                  102, 102, 102, 102, 118, 118, 118, 0, -
                46  46  46  46  00  00  00  00  46  00053                  118, 118, 118, 118, 126, 126, 0, 0, 0, 0, -
                                                                          70, 70, 70, 70, 70, 70, 0, 0, 0, 0, 70, -
                                                                          70, 70, 70

01  0B  06  07  08  07  00  01  01  01  05  04  03  02  00  0005C  P.AAB:  .BYTE   0, 2, 3, 4, 5, 1, 1, 1, 0, 7, 8, 7, 6, -
00  01  01  01  01  01  01  01  01  01  0D  0C  0A  09  01  0006B                  11, 1, 1, 9, 10, 12, 13, 1, 1, 1, 1, 1, -
01  01  01  01  00  00  00  00  00  01  01  01  01  01  01  0007A                  1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0, -
                01  01  01  01  00  00  00  00  01  00089                  0, 0, 1, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, -
                                                                          1

                                                FI_ACT=                 P.AAA
                                                FI_ACT_2=               P.AAB


                     5E        08  C2  00000  FORSSFMT_INTRP1::
                                                SUBL2   #8, SP
                 8D  AB  9F  00003              PUSHAB  -115(CCB)
             01  00  BE  B1  00006              CMPW    a0(SP), #1
                 19  15  0000A              BLEQ    2$
             58  8F  AB  9A  0000C              MOVZBL  -113(CCB), FMT_CODE
             50  80  AF48  9A  00010              MOVZBL  FI_ACT[FMT_CODE], ACT
                 07  59  E9  00015              BLBC    DT_SEEN, 1$
     03          50  06  E1  00018              BBC     #6, ACT, 1$
                 0258  31  0001C              BRW     41$
             00  BE  B7  0001F  1$:            DECW    a0(SP)
                 02B2  31  00022              BRW     52$
             52  80  AB  D0  00025  2$:        MOVL    -128(CCB), P
             58  62  9A  00029  3$:        MOVZBL  (P), FMT_CODE
             58  80  8F  8A  0002C              BICB2   #128, FMT_CODE
             53  FF5F  CF48  9A  00030              MOVZBL  FI_ACT[FMT_CODE], ACT
                 0B  59  E9  00036              BLBC    DT_SEEN, 4$
     07          53  06  E1  00039              BBC     #6, ACT, 4$
         80  AB  52  D0  0003D              MOVL    P, -128(CCB)
                 0233  31  00041              BRW     41$
             58  82  9A  00044  4$:        MOVZBL  (P)+, FMT_CODE
             15  58  CF  00047              CASEL   FMT_CODE, #21, #14
0020     0020     0020     0020  0004B  5$:        .WORD   6$-5$,-
0026     0026     0026     0020  00053                          6$-5$,-
0026     0026     0026     0047  0005B                          6$-5$,-
         0034     0034     0026  00063                          6$-5$,-
                                                                7$-5$,-
                                                                7$-5$,-
                                                                7$-5$,-
                                                                10$-5$,-
                                                                7$-5$,-
                                                                7$-5$,-
                                                                7$-5$,-
                                                                8$-5$,-
                 27  11  00069              8$-5$
                                                BRB     10$
```

```
                    89  AB      82 9B 0006B  6$:    MOVZBW   (P)+, -119(CCB)           0606
                                16 11 0006F         BRB      9$                        0607
                    89  AB      82 9B 00071  7$:    MOVZBW   (P)+, -119(CCB)           0613
                    8B  AB      82 90 00075         MOVB     (P)+, -117(CCB)           0614
                    8C  AB      02 90 00079         MOVB     #2, -116(CCB)             0615
                                08 11 0007D         BRB      9$                        0616
                    89  AB      82 9B 0007F  8$:    MOVZBW   (P)+, -119(CCB)           0622
                    8B  AB      82 B0 00083         MOVW     (P)+, -117(CCB)           0623
                    00  BE      01 B0 00087  9$:    MOVW     #1, @0(SP)                0625
                    96  AB      08 88 0008B         BISB2    #8, -106(CCB)             0626
                              0236 31 0008F         BRW      50$
              28          58    07 E4 00092  10$:   BBSC     #7, FMT_CODE, 14$         0646
                    00  BE      01 B0 00096         MOVW     #1, @0(SP)                0649
              03          53    05 E0 0009A         BBS      #5, ACT, 11$              0650
                              012A 31 0009E         BRW      30$
              03    89  AB      82 9B 000A1  11$:   MOVZBW   (P)+, -119(CCB)           0653
              03          53    04 E0 000A5         BBS      #4, ACT, 12$              0654
                              011F 31 000A9         BRW      30$
              03    8B  AB      82 90 000AC  12$:   MOVB     (P)+, -117(CCB)           0657
              03          53    03 E0 000B0         BBS      #3, ACT, 13$              0658
                              0114 31 000B4         BRW      30$
                    8C  AB      82 90 000B7  13$:   MOVB     (P)+, -116(CCB)           0660
                              010D 31 000BB         BRW      30$                       0646
                    08  AE      82 9A 000BE  14$:   MOVZBL   (P)+, FMT_REPR            0670
                          08   AE 95 000C2         TSTB     FMT_REPR                  0682
                                21 18 000C5         BGEQ     16$
                                82 D0 000C7         MOVL     (P)+, T                   0689
                          50    00 FB 000CA         CALLS    #0, (P)[T]
              00008000  8F      50 D1 000CE         CMPL     T, #32768                 0691
                                04 1E 000D5         BGEQU    15$
                                50 D5 000D7         TSTL     T
                                23 12 000D9         BNEQ     19$
                          7E 44 8F 9A 000DB  15$:   MOVZBL   #68, -(SP)                0694
              00000000G  00    01 FB 000DF         CALLS    #1, FOR$$SIGNAL
                                13 11 000E6         BRB      18$                       0693
                          05 08 AE E9 000E8  16$:   BLBC     FMT_REPR, 17$             0707
                          50    82 9A 000EC         MOVZBL   (P)+, R0                  0709
                                0D 11 000EF         BRB      19$
              05          08 AE 01 E1 000F1  17$:   BBC      #1, FMT_REPR, 18$         0711
                          50    82 3C 000F6         MOVZWL   (P)+, R0                  0713
                                03 11 000F9         BRB      19$
                          50    01 D0 000FB  18$:   MOVL     #1, R0                    0711
                    00  BE      50 B0 000FE  19$:   MOVW     R0, @0(SP)                0682
              03          53    05 E0 00102         BBS      #5, ACT, 20$              0728
                              00C2 31 00106         BRW      30$
              48          08 AE 06 E1 00109  20$:   BBC      #6, FMT_REPR, 23$         0731
                          50    82 D0 0010E         MOVL     (P)+, T                   0737
                          6240  00 FB 00111         CALLS    #0, (P)[T]
                    04  AE      50 D0 00115         MOVL     R0, T
                          0C    58 D1 00119         CMPL     FMT_CODE, #12             0739
        04  AE      04  AE 08   19 12 0011C         BNEQ     21$
                                00 EC 0011E         CMPV     #0, #8, T, T             0743
                                29 13 00125         BEQL     22$
                          7E 44 8F 9A 00127         MOVZBL   #68, -(SP)                0746
              00000000G  00    01 FB 0012B         CALLS    #1, FOR$$SIGNAL
                          04 AE D4 00132         CLRL     T,                          0747
                                19 11 00135         BRB      22$                       0741
```

```
                    00008000   8F      04   AE  D1 00137 21$:       CMPL     T, #32768                      : 0754
                               7E       OF  1F 0013F              BLSSU    22$                              : 
                    00000000G  00       8F  9A 00141              MOVZBL   #68, -(SP)                       : 0757
                               04       01  FB 00145              CALLS    #1, FOR$$SIGNAL                  : 
                               50       01  D0 0014C              MOVL     #1, T                            : 0758
                                    04  AE  D0 00150 22$:         MOVL     T, R0                            : 0763
                                        0D  11 00154              BRB      25$                              : 
                    05          08  AE  02  E1 00156 23$:         BBC      #2, FMT_REPR, 24$                : 0765
                               50       82  3C 0015B              MOVZWL   (P)+, R0                         : 
                                        03  11 0015E              BRB      25$                              : 
                               50       82  9A 00160 24$:         MOVZBL   (P)+, R0                         : 
                          89   AB       50  B0 00163 25$:         MOVW     R0, -119(CCB)                    : 0731
                    60    53       04  E1 00167              BBC      #4, ACT, 30$                          : 0776
                    20    08  AE   05  E1 0016B              BBC      #5, FMT_REPR, 26$                      : 0779
                               50       82  D0 00170              MOVL     (P)+, T                          : 0785
                          6240          00  FB 00173              CALLS    #0, (P)[T]                       : 
                    00008000   8F       50  D1 00177              CMPL     T, #32768                        : 0787
                                        13  1F 0017E              BLSSU    27$                              : 
                               7E       8F  9A 00180              MOVZBL   #68, -(SP)                       : 0790
                    00000000G  00       01  FB 00184              CALLS    #1, FOR$$SIGNAL                  : 
                               50       01  D0 0018B              MOVL     #1, R0                           : 0789
                                        03  11 0018E              BRB      27$                              : 0787
                               50       82  9A 00190 26$:         MOVZBL   (P)+, R0                         : 0797
                          8B   AB       50  90 00193 27$:         MOVB     R0, -117(CCB)                    : 0779
                          8C   AB       02  90 00197              MOVB     #2, -116(CCB)                    : 0798
                    2C    53       03  E1 0019B              BBC      #3, ACT, 30$                          : 0806
                    20    08  AE   04  E1 0019F              BBC      #4, FMT_REPR, 28$                      : 0809
                               50       82  D0 001A4              MOVL     (P)+, T                          : 0815
                          6240          00  FB 001A7              CALLS    #0, (P)[T]                       : 
                    00000100   8F       50  D1 001AB              CMPL     T, #256                          : 0817
                                        13  1F 001B2              BLSSU    29$                              : 
                               7E       8F  9A 001B4              MOVZBL   #68, -(SP)                       : 0820
                    00000000G  00       01  FB 001B8              CALLS    #1, FOR$$SIGNAL                  : 
                               50       01  D0 001BF              MOVL     #1, R0                           : 0819
                                        03  11 001C2              BRB      29$                              : 0817
                               50       82  9A 001C4 28$:         MOVZBL   (P)+, R0                         : 0827
                          8C   AB       50  90 001C7 29$:         MOVB     R0, -116(CCB)                    : 0809
                    04    53       02  E1 001CB 30$:         BBC      #2, ACT, 31$                          : 0845
                          96   AB       08  88 001CF              BISB2    #8, -106(CCB)                    : 
                               03       53  E8 001D3 31$:         BLBS     ACT, 32$                         : 0851
                                    00EF  31 001D6              BRW      50$                                : 
                          0D       00  FDEC CF48 8F 001D9 32$:  CASEB    FI_ACT_2[FMT_CODE], #0, #13       : 0853
      0032   0020    00E8       001C    001E0 33$:  .WORD    34$-33$,-                                      :
      00A3   009C    0078       0050    001E8              50$-33$,-                                         :
      00BB   00B5    00AF       00A9    001F0              35$-33$,-                                         :
                     00E0       00CA    001F8              36$-33$,-                                         :
                                              37$-33$,-                                                      :
                                              39$-33$,-                                                      :
                                              42$-33$,-                                                      :
                                              43$-33$,-                                                      :
                                              44$-33$,-                                                      :
                                              45$-33$,-                                                      :
                                              46$-33$,-                                                      :
                                              47$-33$,-                                                      :
                                              48$-33$,-                                                      :
                                              49$-33$                                                       :
                    3E  DD 001FC 34$:         PUSHL    #62                                                  : 0864
```

```
                                70 11 001FE        BRB   40$
                        92  AB   01 8E 00200  35$:  MNEGB  #1, -110(CCB)                        0894
        90  AB          52       CB A3 00204        SUBW3  -132(CCB), P, -112(CCB)             0895
                        96  AB   08 8A 0020B        BICB2  #8, -106(CCB)                       0896
                                 00B6 31 0020F      BRW    50$                                 0853
                            92  AB 96 00212  36$:   INCB   -110(CCB)                           0908
                            92  AB 9A 00215         MOVZBL -110(CCB), R0                        0909
                FF60 CB40   50  00 BE B0 00219      MOVW   a0(SP), -160(CCB)[R0]               0910
        FF50 CB40          52  FF7C CB A3 00220     SUBW3  -132(CCB), P, -176(CCB)[R0]         0912
                        00  BE   01 B0 00229        MOVW   #1, a0(SP)                          0913
                                 0098 31 0022D      BRW    50$                                 0853
                            50  92 AB 9A 00230 37$: MOVZBL -110(CCB), R0                       0927
                            51  FF60 CB40 3C 00234  MOVZWL -160(CCB)[R0], R1                   0928
                                 51 D7 0023A        DECL   R1
                FF60 CB40        51 B0 0023C        MOVW   R1, -160(CCB)[R0]
                                 51 D5 00242        TSTL   R1
                                 0D 15 00244        BLEQ   38$
                            52  FF50 CB40 3C 00246  MOVZWL -176(CCB)[R0], P                    0933
                            52  FF7C CB C0 0024C    ADDL2  -132(CCB), P                        0932
                                 75 11 00251        BRB    50$
                            92  AB 97 00253  38$:   DECB   -110(CCB)                           0937
                                 70 11 00256        BRB    50$                                 0927
                            52  90 AB 3C 00258 39$: MOVZWL -112(CCB), P                        0952
                            52  FF7C CB C0 0025C    ADDL2  -132(CCB), P
                        92  AB   01 8E 00261        MNEGB  #1, -110(CCB)                       0953
                                 5A D5 00265        TSTL   EL_SIZE                             0955
                                 5F 13 00267        BEQL   50$
        5A       96  AB          03 E0 00269        BBS    #3, -106(CCB), 50$
                                 3C DD 0026E        PUSHL  #60
        00000000G  00            01 FB 00270  40$:  CALLS  #1, FOR$$SIGNAL_STO                 0958
                                 58 D4 00277  41$:  CLRL   FMT_CODE                           0959
                                 00F4 31 00279      BRW    66$                                 0957
                        88  AB   89 AB 90 0027C 42$:MOVB   -119(CCB), -120(CCB)               0974
                                 45 11 00281        BRB    50$                                 0853
                        94  AB   01 8A 00283  43$:  BICB2  #1, -108(CCB)                       0984
                                 3F 11 00287        BRB    50$                                 0853
                        94  AB   01 88 00289  44$:  BISB2  #1, -108(CCB)                       0994
                                 39 11 0028D        BRB    50$                                 0853
                        93  AB   01 88 0028F  45$:  BISB2  #1, -109(CCB)                      1004
                                 33 11 00293        BRB    50$                                 0853
                        93  AB   01 8A 00295  46$:  BICB2  #1, -109(CCB)                      1014
                                 2D 11 00299        BRB    50$                                 0853
                            50  89 AB 3C 0029B 47$: MOVZWL -119(CCB), R0                      1025
                            50  BC AB C0 0029F      ADDL2  -68(CCB), R0
                        B0  AB   FF A0 9E 002A3     MOVAB  -1(R0), -80(CCB)
                                 1E 11 002A8        BRB    50$                                 0853
                            50  89 AB 3C 002AA 48$: MOVZWL -119(CCB), R0                      1035
                        B0  AB   50 C2 002AE        SUBL2  R0, -80(CCB)
                        BC  AB   B0 AB D1 002B2     CMPL   -80(CCB), -68(CCB)                  1037
                                 0F 1E 002B7        BGEQU  50$
                        B0  AB   BC AB D0 002B9     MOVL   -68(CCB), -80(CCB)                  1039
                                 08 11 002BE        BRB    50$                                 0853
                            50  89 AB 3C 002C0 49$: MOVZWL -119(CCB), R0                      1054
                        B0  AB   50 C0 002C4        ADDL2  R0, -80(CCB)
        03       53             01 E0 002C8  50$:   BBS    #1, ACT, 51$                        1073
                                 FD5A 31 002CC      BRW    3$
                        8F  AB   58 90 002CF  51$:   MOVB   FMT_CODE, -113(CCB)                1079
```

FOR$$FMT_INTRP  Fortran Format Statement Interpreter
2-037

E 15
16-Sep-1984 00:25:18    VAX-11 Bliss-32 V4.0-742
14-Sep-1984 12:32:00    [FORRTL.SRC]FORFMTINT.B32;1

Page 29
(7)

```
                              80  AB          52 D0 002D3         MOVL    P, -128(CCB)                           ; 1080
                                  29          58 D1 002D7 52$:    CMPL    FMT_CODE, #41                          ; 1089
                                              03 1E 002DA         BGEQU   53$
                                           0091 31 002DC         BRW     66$
               0C          29          58 CF 002DF 53$:    CASEL   FMT_CODE, #41, #12                      ; 1093
     0026    0037        0020        001A    002E3 54$:    .WORD   55$=54$,-
     008A    008A        008A        0037    002EB                        56$=54$,-
     005F    005F        005F        008A    002F3                        59$=54$,-
                                      005F    002FB                        57$=54$,-
                                                                           59$=54$,-
                                                                           65$=54$,-
                                                                           65$=54$,-
                                                                           65$=54$,-
                                                                           65$=54$,-
                                                                           62$=54$,-
                                                                           62$=54$,-
                                                                           62$=54$,-
                                                                           62$=54$

                              89  AB          5A B0 002FD 55$:    MOVW    EL_SIZE, -119(CCB)                     ; 1103
                                              6A 11 00301         BRB     65$
                              89  AB          02 B0 00303 56$:    MOVW    #2, -119(CCB)                          ; 1111
                                              64 11 00307         BRB     65$
                                  04          5A D1 00309 57$:    CMPL    EL_SIZE, #4                            ; 1120
                                              06 1E 0030C         BGEQU   58$
                              89  AB          07 B0 0030E         MOVW    #7, -119(CCB)
                                              59 11 00312         BRB     65$
                              89  AB          0C B0 00314 58$:    MOVW    #12, -119(CCB)
                                              53 11 00318         BRB     65$
                        50          5A          03 78 0031A 59$:    ASHL    #3, EL_SIZE, R0                       ; 1131
                                    50          02 C0 0031E         ADDL2   #2, R0
                                    50          03 C6 00321         DIVL2   #3, R0
                                              50 D6 00324         INCL    R0
              0000FFFF  8F          50 D1 00326         CMPL    R0, #65535
                                              05 15 0032D         BLEQ    60$
                        50    FFFF  8F 3C 0032F         MOVZWL  #65535, R0
                                    07          50 D1 00334 60$:    CMPL    R0, #7
                                              03 18 00337         BGEQ    61$
                                    50          07 D0 00339         MOVL    #7, R0
                              89  AB          50 B0 0033C 61$:    MOVW    R0, -119(CCB)
                                              2B 11 00340         BRB     65$
                                  04          5A D1 00342 62$:    CMPL    EL_SIZE, #4                            ; 1146
                                              1E 13 00345         BEQL    64$
                                  08          5A D1 00347         CMPL    EL_SIZE, #8                            ; 1153
                                              0A 12 0034A         BNEQ    63$
                    89  AB 02100019 6F D0 0034C         MOVL    #34603033, -119(CCB)                   ; 1156
                                              17 11 00354         BRB     65$                                   ; 1143
                                  10          5A D1 00356 63$:    CMPL    EL_SIZE, #16                          ; 1160
                                              0A 12 00359         BNEQ    64$
                    89  AB 0321002A 8F D0 0035B         MOVL    #52494378, -119(CCB)                   ; 1163
                                              08 11 00363         BRB     65$                                   ; 1143
                    89  AB 0207000F 8F D0 00365 64$:    MOVL    #34013199, -119(CCB)                   ; 1170
                                              58 14 C2 0036D 65$:    SUBL2   #20, FMT_CODE                        ; 1185
                                              5E 0C C0 00370 66$:    ADDL2   #12, SP                              ; 1196
                                              05 00373         RSB
```

; Routine Size:  884 bytes,    Routine Base:  _FOR$CODE + 0092

```
; 1138           1197  1
; 1139           1198  1 END                                        ! End of module FOR$$FMT_INTRP
; 1140           1199  1
; 1141           1200  0 ELUDOM
```

```
;                          PSECT SUMMARY

;      Name                      Bytes                    Attributes

; _FOR$CODE                       1030  NOVEC,NOWRT,  RD , EXE, SHR, LCL, REL, CON, PIC,ALIGN(2)
```

```
;                     Library Statistics

;                                        -------- Symbols --------    Pages      Processing
;      File                               Total  Loaded   Percent    Mapped        Time

; _$255$DUA28:[SYSLIB]STARLET.L32;1        9776       0        0        581      00:01.1
; _$255$DUA28:[FORRTL.OBJ]FORLIB.L32;1      711     211       29         52      00:00.6
; _$255$DUA28:[FORRTL.OBJ]RTLLIB.L32;1       36       0        0          8      00:00.1
```

```
;                     COMMAND QUALIFIERS

;      BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/NOTRACE/LIS=LIS$:FORFMTINT/OBJ=OBJ$:FORFMTINT MSRC$:FORFMTINT/UPDATE=(ENH$:FORFMTINT
;      )

; Size:           922 code + 108 data bytes
; Run Time:         00:29.5
; Elapsed Time:     01:23.3
; Lines/CPU Min:      2441
; Lexemes/CPU-Min: 23045
; Memory Used:   386 pages
; Compilation Complete
```

FORFIND
LIS

FORFMTINT
LIS

FOREXITHA
LIS

FORENODEF
LIS

FORENCOMF
LIS

FORIDATE
LIS

FORENDFIL
LIS

FORERRSNS
LIS

FOREXIT
LIS

FORERROR
LIS

FORFMTOP
LIS

FORENCOMO
LIS

FORINIDES
LIS